Faster BERT-based re-ranking through Candidate Passage Extraction

Kyle Reed
Department of Computer Science
University of Sheffield
United Kingdom
kgreed1@sheffield.ac.uk

Harish Tayyar Madabushi
School of Computer Science
University of Birmingham
United Kingdom
harish@harishtayyarmadabushi.com

Abstract

Most modern information retrieval systems employ a multi-step approach to retrieving documents relevant to a query, first retrieving a set of candidate documents before re-ranking the candidates. The most effective methods of re-ranking use a transformer-based classifier to score documents. Since many documents exceed the input length of transformers, they are split into passages and each passage is classified independently, aggregating the scores for an overall document score. As transformers are slow due to their quadratic attention mechanism, we investigate whether extracting only the most promising passages from documents as input for the classifier can alleviate slow performance on longer documents at inference time while maintaining comparable performance. We explore three methods of passage extraction and find these approaches prove effective, performing comparably to the state-of-the-art while significantly reducing the run-time, with the best results coming from a graph-based passage-ranking algorithm.

1 Introduction

Information retrieval is the task of identifying documents that contain information relevant to some user requirement, often a query. Since document collections are often large, retrieval usually consists of two stages: a cheap initial retrieval of k documents, followed by a more involved re-ranking. The Deep Learning Track at TREC 2020 provides an opportunity to investigate different approaches to information retrieval with a large scale dataset. There are two tasks - passage ranking and document ranking based on their relevance to a query, each with two sub-tasks; end-to-end ranking and re-ranking, mimicking the traditional IR pipeline. We work on the document re-ranking sub-task, where we are tasked with re-ranking a set of 100 documents that have already been retrieved so that the documents most relevant to a given query are placed highest.

This work builds on the work by Yan et al. (2019) and Chen et al. (2019) who approach the re-ranking task using BERT (Devlin et al., 2019) to classify the relevance of a document's constituent passages to a query, aggregating these scores to get an overall document score. Document scores are taken to be the maximum score of any passages contained in that document, working under the assumption that only one part of the document needs be relevant to a query for that document to be relevant (Yan et al., 2019; Chen et al., 2019). This approach is effective, but the slow performance of transformers at inference-time, combined with the large number of passages in some document could result in the passage classification being a bottle-neck in a real retrieval system. We aim to solve this problem by limiting the number of passages to be classified by identifying a subset of potentially relevant passages ahead of scoring them, introducing a "passage extraction" stage into the ranking pipeline.

Our overall approach to re-ranking is as follows: from each of the top-k documents, extract five candidate passages we expect to be relevant to the query (resulting in a positive document classification). Then, we use a classifier based on the sentence embeddings generated by BERT, fine-tuned to classify the relevance of a passage to a given query to score each of our passages. Finally, we re-rank the documents under the assumption that a documents relevance is determined by its most relevant passage, using the maximum passage score as the document score (Yan et al., 2019; Chen et al., 2019). We experiment with three different methods for passage extraction. These methods are: a) taking document spans around occurrences of query keywords, b) ranking passages based on their semantic similarity to the query, and c) adapting the TextRank (Mihalcea and Tarau, 2004) algorithm to find the passages that are most representative of the document. Our methods perform (NDCG @ 10: 0.5781/0.5830/0.5949) comparably to using the unfiltered set of passages (NDCG @ 10: 0.5937), while significantly reducing the cost of inference, a key consideration in time-sensitive real-world systems. We hypothesise that passage extraction will have a favourable effect on the run-time of a re-ranking approach, while maintaining performance.

To ensure reproducible and so as to enable other researchers to build upon this work, we make the program code associated with this work available publicly [1].

2 Related Work

The most effective approaches of the 2019 track leveraged neural-network language-models (Craswell et al., 2019), fine-tuning a BERT passage classifier and splitting a document up into overlapping passages, taking the max passage score as the document score (Yan et al., 2019; Chen et al., 2019). While BERT proves to perform well on this task, there are shortcomings associated with its application in real-world systems. The most prohibitive of these in this task are its demands at inference time (Wang et al., 2020). This cost has the potential to hamper the adoption of BERT models in real-world, time-sensitive systems (Sanh et al., 2019; Ganesh et al., 2020; Wang et al., 2020).

There have been several attempts to overcome this limitation of BERT, by altering the attention mechanism or by compressing the model. Wang et al. (2020) and Beltagy et al. (2020) use linear self-attention mechanisms to improve inference-time efficiency for longer sequences. Sanh et al. (2019) use knowledge distillation to train a general purpose pre-trained BERT called DistilBERT, that is 40% smaller and 60% faster at inference time, maintaining 97% of performance. However, the success of these efforts is most noticeable on significantly longer sequence lengths, or comes with a drop in performance, which is not appropriate for every task.

3 Methodology

Here we outline our methods for finding candidate relevant passages. We establish a baseline of two passages using the title and the first 384 tokens of the body as the only passages. For each other method we extract five passages in total. We find that the title is an extremely strong predictor for document relevance, so use it as one of these passages, and also append it to the start of every other passage. To maintain coherence, our passages are a list of sentences as they appear in the document.

3.1 Keyword windows

Our first method for passage selection is based on an understanding of BERT in passage-ranking contexts and our observations of the queries. Qiao et al. (2019) note that: “exact match terms play an important role in BERT; we found many of the influential terms in BERT are those that appear in the question or close paraphrases”, and also “there are a few terms in each document that determine the majority of BERT’s ranking scores”. We leverage this by extracting the keywords from the query and looking for exact matches in the document body, hypothesising that such passages would be most likely to contain information relevant to the query. For example, from the query “who is Serena Williams tennis”, we would extract the terms “tennis” and “Serena Williams”, and use passages centring around document occurrences of these terms as our input.

3.2 Query-passage similarity

Our next method for passage selection makes use of static word embeddings as a less accurate but faster method of finding candidate relevant passages identify the passages that are most relevant to the query, similar to prior work at the document level (Nalisnick et al., 2016). We first create passages by splitting the document body into overlapping passages, where we use the best passage length found from our keyword windows experimentation in Section 3.1. We remove stop-words and out-of-vocabulary words

[1] github.com/kylereed96/trec-dl-2020

from the passages. Given q_i and p_j as the embedding vectors for the i^{th} and j^{th} terms of the query and passage respectively, we then obtain a score for each passage as

$$f(Q,P) = \frac{1}{|Q|} \sum_{q_i \in Q} \frac{q_i^T \bar{P}}{||q_i||\,||\bar{P}||} \tag{1}$$

where, $\bar{P} = \frac{1}{|P|} \sum_{p_j \in D} \frac{p_j}{||p_j||}$. We experiment with three types of embedding, GloVe (Pennington et al., 2014) word2Vec embeddings (Mikolov et al., 2013), and SIF embeddings (Arora et al., 2019) We find that GloVe performs the best, though not by a significant margin, with no improvement from using longer embeddings.

3.3 PassageRank

The final method we use for passage extraction is a variant on TextRank (Mihalcea and Tarau, 2004). TextRank represents passages of text as vertices of a graph, with similarity between these passages being the edges. With a directed graph $G = (V, E)$ that has vertices V and edges E, where $In(V_i)$ is the set of vertices pointing to vertex V_i and $Out(V_i)$ is the set of vertices that vertex V_i points to, the score of V_i is given by

$$WS(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} \frac{w_{ji}}{\sum_{V_k \in Out(V_j)} w_{jk}} WS(V_j) \tag{2}$$

where d is a dampening factor usually set to 0.85 (Brin and Page, 1998) and w_{ji} is the weight between edges i and j. Starting from arbitrary values, scores are iteratively re-calculated until convergence.

We weight the edges between vertices using two metrics: the original TextRank metric, and the cosine similarity of our GloVe passage embeddings in eq. (1). The TextRank metric for the similarity between two passages containing words w_i is given by

$$Similarity(P_i, P_j) = \frac{|\{w_k | w_k \in P_i \& w_k \in P_j\}|}{log(|P_i|) + log(|P_j|)} \tag{3}$$

We find the best results for passages containing 12 sentences (tuned as a hyper-parameter), ranked using cosine similarity of GloVe passage embeddings as the edge weightings.

3.4 BERT passage classifier

We fine-tune BERT with a passage classification objective using the passage-level dataset provided, with a max sequence length of 384. We find best results using standard hyper-parameters. We use this model to give each passage (selected using one of the methods described above) a score determining its relevance to the given query and use the highest of these as the document score.

4 Experiment and Results

Each of our methods make a substantial improvement over the initial ranking. We set up two baselines: title_body (consisting of the title and start of the body), and all_passages (the title and the full body split into overlapping passages of length 12 sentences and stride 6 sentences). Our passage-extraction methods all perform comparably to these baselines, with best results coming from PassageRank. This suggests that a document's most representative passages are the best for determining a documents relevance to a query. Our title_body baseline also proves to be very strong.

Run ID	Group	Run Description	Subtask	NDCG@10	MAP	MRR
initial_ranking	TREC	no re-ranking	re-rank	0.4980	0.3541	0.7981
all_passages	UoB	full text (baseline)	re-rank	0.5937	0.4094	0.8630
title_body	UoB	title & body (baseline)	re-rank	0.5779	0.3755	0.9130
uob_runid1	UoB	keyword windows	re-rank	0.5781	0.3786	0.8852
uob_runid2	UoB	query-passage similarity	re-rank	0.5830	0.3976	0.9100
uob_runid3	UoB	PassageRank	re-rank	0.5949	0.3948	0.9259

Table 1: Ranking performance of baseline and submitted runs in document re-ranking task

4.1 Statistical significance

We run statistical significance tests to determine: a) whether the proposed methods, based on re-ranking, are genuine improvements over the initial retrieval; and b) whether any method noticeably outperforms another. Urbano et al. (2013) note that traditional IR metrics do not lend themselves to traditional significance tests due to their violation of many necessary assumptions but perform empirical analysis to conclude that, in practice, they are still appropriate. Furthermore, the t-test is the safest test and the Wilcoxon test gives the most exact results. We use the t-test and Wilcoxon, noting that our small sample size favours Wilcoxon. The null hypothesis is that our rankings come from the same distribution. We include these results in Table 2. All of our methods of re-ranking significantly outperform the initial ranking ($p < 0.01$) on the Wilcoxon test, but do not pass the t-test ($p > 0.05$). None of the other methods for re-ranking significantly outperform each other, including our title_body baseline.

	title_body	all_passages	uob_runid1	uob_runid2	uob_runid3
initial_ranking	*0.0065* **0.0980**	*0.0010* **0.0607**	*0.0071* **0.1119**	*0.0057* **0.0929**	*0.0011* **0.0510**
title_body	-	*0.1311* **0.7354**	*0.7572* **0.9966**	*0.8307* **0.9127**	*0.2089* **0.7078**
all_passages	-	-	*0.5360* **0.7495**	*0.2369* **0.8270**	*0.9777* **0.9807**
uob_runid1	-	-	-	*0.9226* **0.9197**	*0.4645* **0.7238**
uob_runid2	-	-	-	-	*0.2247* **0.8030**

Table 2: p-values for *Wilcoxon and **t-test***

4.2 Runtime

We include a comparison of the run-times of the different methods in Table 3, including the fraction of time the approach takes compared to all_passages, the baseline consisting of the whole document body split into overlapping passages.

Method	Run Description	Preparation (s)	Inference (s)	Total (s)	Fraction of all_passages
all_passages	full text (baseline)	1503 (0309)	6706 (1508)	8209 (1817)	-
title_body	title & body (baseline)	0165 (0098)	1178 (0232)	1343 (0330)	0.1636 (0.1816)
uob_runid1	keyword windows	0679 (0112)	2927 (0604)	3606 (0716)	0.4393 (0.3941)
uob_runid2	query-passage similarity	1264 (0237)	2928 (0603)	4192 (0840)	0.5107 (0.4623)
uob_runid3	PassageRank	4015 (0673)	2924 (0601)	6939 (1274)	0.8453 (0.7011)

Table 3: Run-time comparison of methods 2020 (2019)

Each of our methods noticeably reduces the overall run-time from classifying all passages. The effect is more noticeable on the 2019 set than the 2020 set, which can be explained by variations in the passage distribution. The title-body baseline is around 6 times faster and keyword windows and query-passage similarity methods for passage extraction make the re-ranking process 2 - 2.5 times faster.

5 Conclusion

The aim of this work was to determine whether a document re-ranking pipeline could achieve similar results to the state-of-the-art by incorporating a passage-selection stage prior to using a transformer-based classifier, alleviating the impact of the slow inference of transformers. We train a BERT passage classifier model that determines the relevance of a passage to a query. We then introduced three methods for extracting input passages that achieve similar performance to state-of-the-art on the document re-ranking task, while significantly reducing run-time.

We conclude that performance is not degraded by identifying a subset of potentially relevant passages prior to the transformer classification stage, but run-time is positively affected. We further note that using just the document title and the start of the body as passages forms a very strong baseline while being an order of magnitude faster.

References

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2019. A simple but tough-to-beat baseline for sentence embeddings. In *5th International Conference on Learning Representations, ICLR 2017*, January.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.*, 30(1–7):107–117, April.

Xuanang Chen, Canjia Li, Ben He, and Yingfei Sun. 2019. UCAS at TREC-2019 deep learning track. In *Proceedings of the Twenty-Eighth Text REtrieval Conference*. National Institute of Standards and Technology.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M. Voorhees. 2019. Overview of the TREC 2019 deep learning track. *The Twenty-Eighth Text REtrieval Conference (TREC 2019) Proceedings*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Prakhar Ganesh, Yao Chen, Xin Lou, Mohammad Ali Khan, Yin Yang, Deming Chen, Marianne Winslett, Hassan Sajjad, and Preslav Nakov. 2020. Compressing large-scale transformer-based models: A case study on BERT. *arXiv preprint arXiv:2002.11985*.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain, July. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013*.

Eric Nalisnick, Bhaskar Mitra, Nick Craswell, and Rich Caruana. 2016. Improving document ranking with dual word embeddings. In *Proceedings of the 25th International Conference Companion on World Wide Web*, page 83–84. International World Wide Web Conferences Steering Committee.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, October.

Yifan Qiao, Chenyan Xiong, Zhenghao Liu, and Zhiyuan Liu. 2019. Understanding the behaviors of BERT in ranking. *arXiv preprint arXiv:1904.07531*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Julián Urbano, Mónica Marrero, and Diego Martín. 2013. A comparison of the optimality of statistical significance tests for information retrieval evaluation. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, page 925–928, New York, NY, USA. Association for Computing Machinery.

Sinong Wang, Belinda Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*.

Ming Yan, Chenliang Li, Chen Wu, Bin Bi, Wei Wang, Jiangnan Xia, and Luo Si. 2019. IDST at TREC 2019 deep learning track: Deep cascade ranking with generation-based document expansion and pre-trained language modeling. In *Proceedings of the Twenty-Eighth Text REtrieval Conference*. National Institute of Standards and Technology.